Proceedings of the 2007 Winter Simulation Conference
S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds.

A SIMULATION FRAMEWORK FOR ENERGY EFFICIENT DATA GRIDS

Ziliang Zong

Department of Computer Science and Software Engineering
Auburn University, AL 36849-5347

Kiranmai Bellam

Department of Computer Science and Software Engineering
Auburn University, AL 36849-5347

Xiao Qin

Department of Computer Science and Software Engineering
Auburn University, AL 36849-5347

Yiming Yang

Intel Corporation
Rio Rancho, NM 87124-1025

Xiaojun Ruan

Department of Computer Science and Software Engineering
Auburn University, AL 36849-5347

Adam Manzanares

Department of Computer Science and Software Engineering
Auburn University, AL 36849-5347

ABSTRACT

High performance data grids are increasingly becoming popular platforms to support data-intensive applications. Reducing high energy consumption caused by data grids is a challenging issue. Most previous studies in grid computing focused on performance and reliability without taking energy conservation into account. As such, designing energy-efficient data grid systems became highly desirable. In this paper, we proposed a framework to simulate energy-efficient data grids. We presented an approach to integrate energy-aware allocation strategies into energy-efficient data grids. Our framework aims at simulating a data grid that can conserve energy for data-intensive applications running on data grids.

1 INTRODUCTION

A data grid is a collection of geographically dispersed storage resources over either local or wide area network. The goal of data grid system is to provide a large virtual storage framework with unlimited power through collaboration among individuals, institutions, and resources (Qin and Jiang 2005). With the rapid growth of requirements for high performance data-intensive computing, data grids are increasingly becoming popular platforms to support data-intensive applications.

Although various research issues in data grids like performance, security, availability, middleware design, fault tolerance have been addressed in the past ten years, the issue of energy conservation in data grids has not received much attention. According to EUN (Energy User News) (Moore 2002), the power requirements of today's data centers in a data grid range from 75 W/ft2 to 150-200 W/ft2 and will increase to 200-300 W/ft2 in the nearest future. The new data center capacity projected for 2005 would require approximately 40TWh ($4B at $100 per MWh) per year to run 24x7 unless they become more efficient (Chase and Doyle 2001). The next supercomputing center in Seattle has been forecasted to increase the city's power demands by 25% (Bryce 2000). Huge-energy consumption will simultaneously cause serious environmental pollution. Based on the data from EPA, generating 1 kWh of electricity in the United States results in an average 1.55 pounds (lb) of carbon dioxide (CO2) emissions. Therefore, it is highly desirable to design economically attractive and environmentally friendly framework for supercomputing platforms like data grids.

Our long term goal is to develop energy-efficient data grids to provide significant energy savings for data-intensive applications running on computational grids. Basically, CPUs, networks, disks, and cooling system are four major power consumers in a data grid. Many literatures have shown that reducing energy dissipation in interconnections is critical. In this paper, we proposed a framework to simulate an energy-efficient data grid. The framework facilitates fundamental functionalities, including computing objects, network objects, disk objects, and scheduling objects. A major contribution of this research is that the simulation framework can be used to predict

the energy consumption and performance of various scheduling schemes for data grids. To validate the framework, we implemented an energy-aware scheduling algorithm in the simulation framework. The energy-aware scheduling algorithm consists of two phases. The first phase is geared to minimize communication overheads among parallel data-intensive tasks by duplicating communication-intensive tasks. In the second phase, the algorithm judiciously allocates grouped data-intensive tasks to most energy-saving computing nodes to optimize the energy conservation of the whole data grid system.

The rest of the paper is organized as follows. Section 2 depicts a general system architecture. In Section 3, we illustrated the simulation framework and explained the function of each module and their relationships. Section 4 describes the task analyzer in the framework. Finally, Section 5 is the concluding remarks and future research directions.

2 SYSTEM ARCHITECTURE

A data grid can be envisioned as a complicated distributed system, which consists of the following four major layers: application layer, middleware layer, resource layer and network layer. Fig. 1 depicts the four layers and their relations for data grids. It is worth noting that the simulation framework is constructed in accordance with the system architecture outlined in Figure 1.

Network layer is the base of all the other layers, which assures the connectivity for the resources in a data grid. On top of it lies the resource layer, made up of the actual resources that are part of the data grid, such as computing nodes, storage systems, electronic data catalogues, and even satellites or other instruments, which can be connected directly to the network. The middleware layer lies at the heart of the data grid system, because this layer contains various intelligent modules like resource brokers, security access controllers, task analyzers, task schedulers, communication services, information services, and reliability controllers. These modules are seamlessly

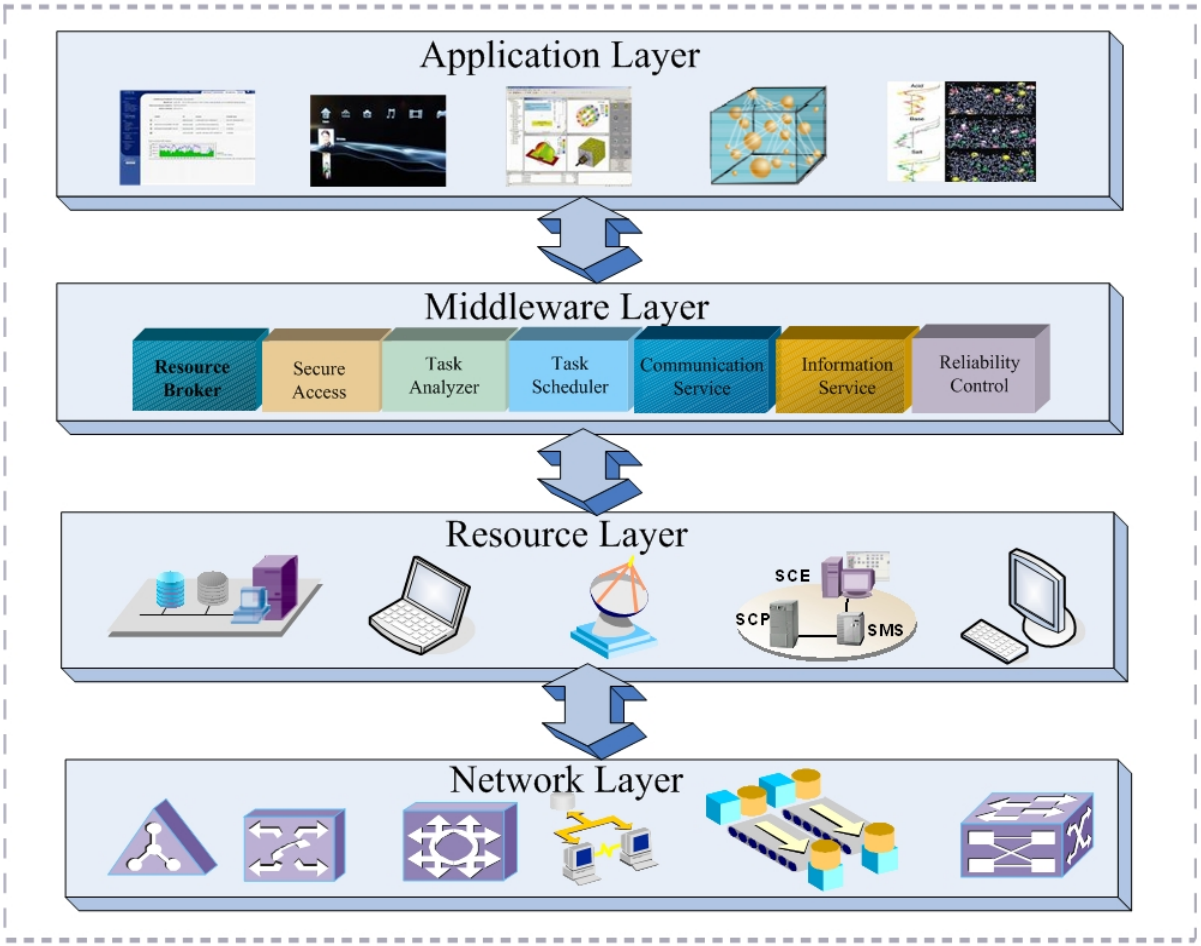


Figure 1: A system architecture for data grids

integrated to efficiently support a variety of computing, storage, and data resources. On the very top of the system architecture is the application layer, which includes all kinds of user data-intensive applications varying from scientific, engineering, business or financial domains. The application layer also consists of portals and development toolkits supporting data-intensive applications.

There are seven modules in the middle layer which are collaborating with each other to assure the performance of a data grid system. In particular, a resource broker is the first gate facing towards the application layer. Once users install the corresponding software supporting data grids, the users can connect to the data grid by submitting their data requests to the resource broker. The security module ensures that the users are authorized to link to the data grid to access data resources. Once access control is granted, a task analyzer can partition a large data-intensive job into a number of small tasks. Dependencies among these small data-intensive tasks will be analyzed. Next, a task scheduler will allocate tasks to distributed computing and storage resources based on different scheduling strategies. In a large-scale data grid, the efficiency of task scheduler noticeably effects the performance of the entire data grid system. One of the main contributions of this study is to implement an efficient energy-aware scheduler into the simulation framework. In addition, a communication service module is responsible for supporting services such as remote function calls. An information service module keeps track of all the detailed executing information pertinent to the tasks running on different computing resources. Last but not least, a reliable control module can guarantee the stability of the data grid system. For example, the reliable control module could reject a scheduling decision made by the task scheduler if allocated computing and storage resource failed to meet a certain reliability requirements.

3 THE SIMULATION FRAMEWORK

Data grids are complex multivariate environments, which are made up of numerous grid entities that need to be automatically managed. In order to make coherent and coordinated use of ubiquitous and heterogeneous data and storage resources, resource management is a centerpiece in the simulation framework. In this section, we present a framework to simulate data grids that are energy efficient

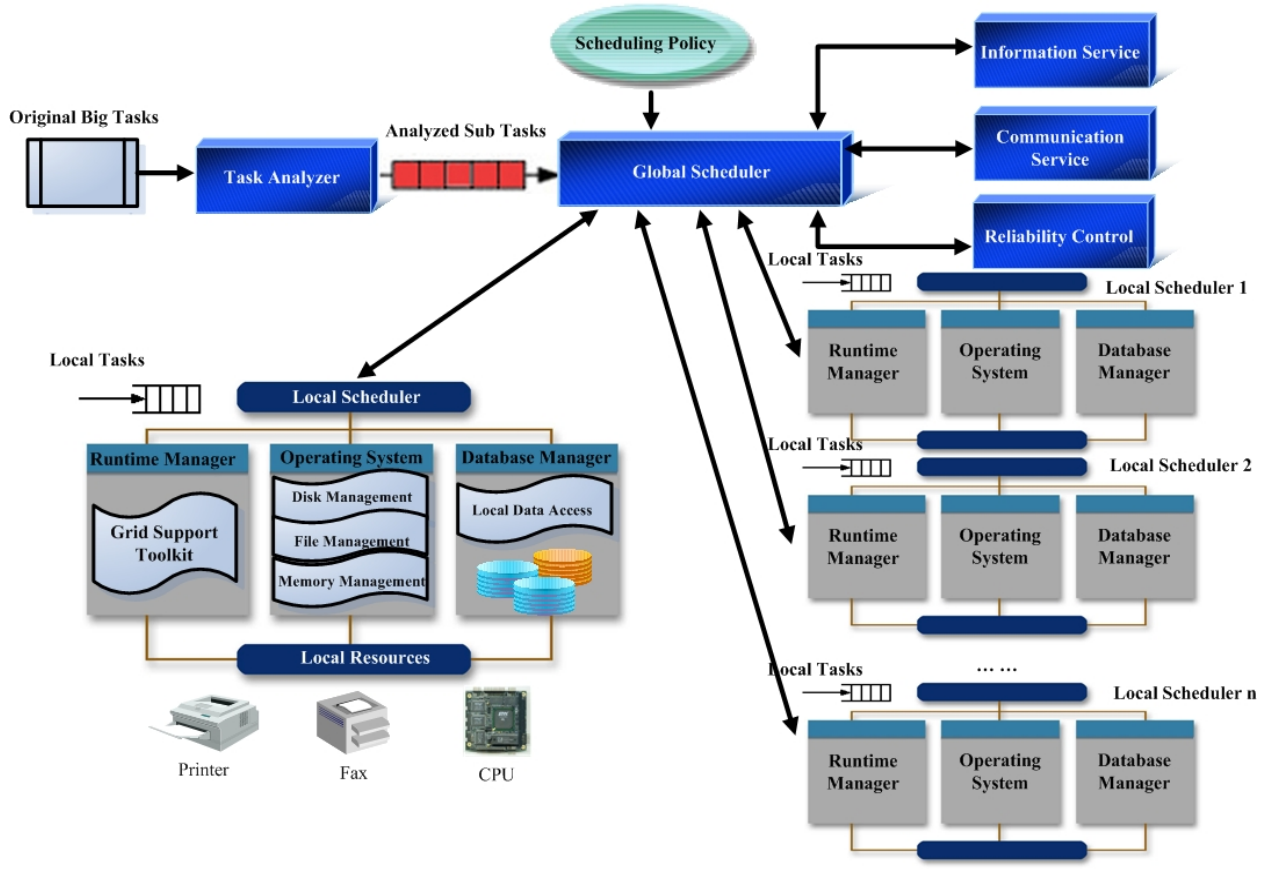


Figure 2: The simulation framework for energy-efficient data grids

in nature. In general, a data grid should energy efficiently handle two important components in the system: storage resources and data-intensive jobs. A data grid has to address the following issues. First, it is of paramount importance to find available storage resources within a short period of time. Second, it must allocate data-intensive jobs to available resources. Fig. 2 illustrates the simulation framework for data grids. In this framework, the global level scheduler (or grid level scheduler) coordinates multiple local scheduling or helps to select the most appropriate resources for a job among different possible resources.

Typically, a global level scheduler itself has no direct control over computing and storage resources. Therefore, the global level scheduler has to communicate with and appropriately trigger several local level schedulers to complete data-intensive tasks submitted by users. Those local level schedulers either control resources directly or have certain access to their local resources. The global level scheduler is also responsible for collaborating with other important supportive middleware like information services, communication services, and reliability control modules.

As long as the grid scheduling module collected all the information of currently available computing and storage resources, it can judiciously choose target recourses based on its scheduling policy and allocate the tasks analyzed by the task analyzer to these chosen resources for parallel execution. Figure 3 shows the job scheduling flow in the simulation framework. During the process of execution, the results collector will periodically check the randomly returned sub results and transfer these sub results to Grid level scheduler. The scheduler in the framework passes the latest information to all tasks, which can guarantee that the tasks with dependency could immediately be executed once they get the necessary sub results.

In order to deal with resource heterogeneities and support our energy-efficient scheduling policy, the grid level task scheduler has the following important attributes:

- **Withdraw allocations**

The target resources could be taken back by a local administrator and allocated to execute data-intensive tasks with higher priorities. In this case, the scheduler must be able to withdraw its allocations and reallocate corresponding tasks to other available computing and storage sources.

- **Allocation Migration**

This attribute indicates that any task can be terminated in one computing node and the corresponding data are packed such that the data-intensive task can be migrated to another computing node in the system. Then, the task can be restarted in the new location. Note that there is no need to re-execute the task from ground up when storage resources become unavailable due to permanent or transient failures.

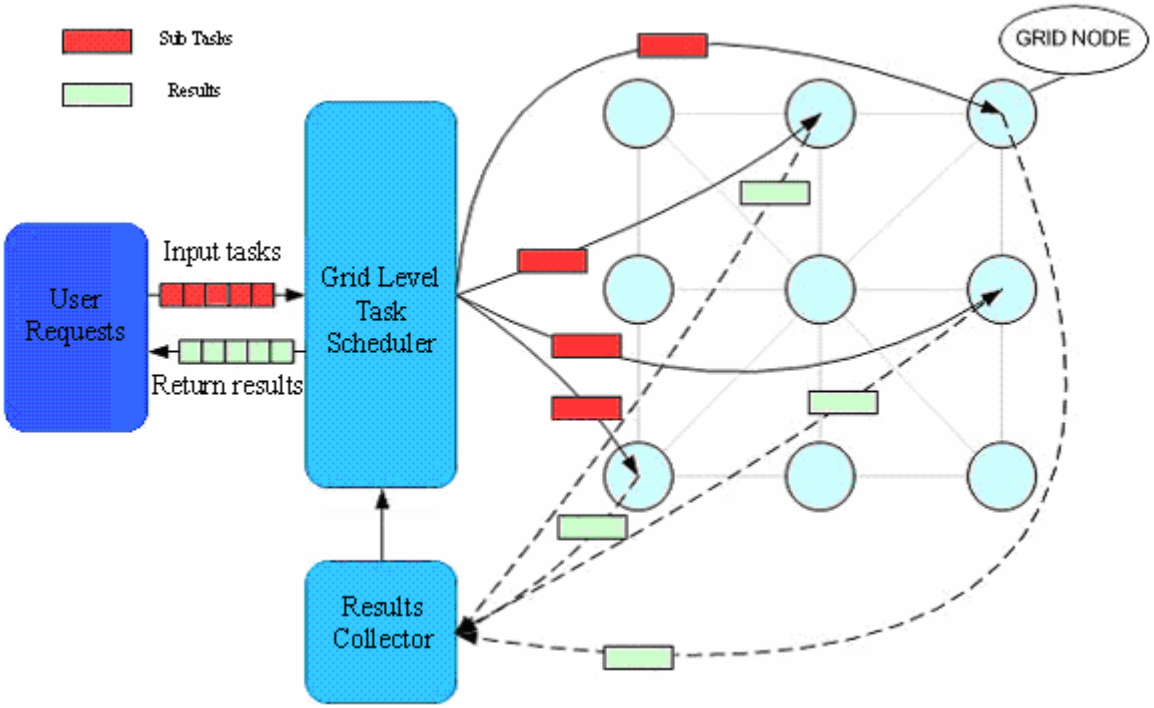


Figure 3: Job scheduling in the framework

- **Exclusive allocations**

Different computing and storage resources might have particular preference or exclusiveness for various type of tasks, meaning that a computing node, which offers task t_i a shorter execution time, does not necessarily run faster for another task t_j. Even worse, some computing nodes are exclusive to specific type of tasks. The scheduler has to avoid the conflicts between tasks and resources.

- **Tentative allocations**

To make an ideal allocation decision, the scheduler needs to calculate and compare allocation cost by tentatively allocating tasks to different available computing and storage resources. The scheduler will have to be able to easily complete revocation of tentative allocations.

- **Dependent task allocations**

The scheduler has to take task dependencies into account. In our framework, the task analyzer provides detailed information regarding data-intensive tasks to the scheduler. The scheduler is always trying to allocate tasks with high dependencies to the same resources in order to reduce communication overheads.

4 TASK ANALYZER

The task analyzer aims at analyzing task characteristics and task dependencies. In addition, the task analyzer is responsible for estimating the execution time of each data-intensive task in accordance with its task type or other information provided by users. In our framework, data-intensive tasks with dependencies are modeled by *Directed Acyclic Graphs (DAGs)*. Throughout this paper, a collaborative application is specified as a pair, i.e, *(T, E)*, where $T = \{t_1, t_2, ..., t_n\}$ represents a set of parallel tasks, E is a set of weighted and directed edges representing communication costs among tasks, e.g., $(t_i, t_j) \in E$ is a message transmitted from task t_i to t_j. Precedence constrains of the parallel tasks are represented by all edges in set E. Communication time spent in delivering a message $(t_i, t_j) \in E$ from task t_i on node p_u to t_j on p_v is determined by s_{ij}/b_{uv}, where s_{ij} is the message's data size and b_{uv} is the transmission rate of a link connecting node p_u and p_v. The execution times of task t_i running on a set of heterogeneous computing nodes are modeled by a vector, i.e., $c_i = \left(c_i^1, c_i^2, \cdots, c_i^m\right)$, where c_i^j represents the execution time of t_i on the jth computing node. If task t_i cannot be executed on node p_j, the corresponding execution time c_i^j in the vector c_i is marked as ∞. We define a task as an entry task if it does not have any predecessor tasks and; similarly, a task is called an exit task if there is no task

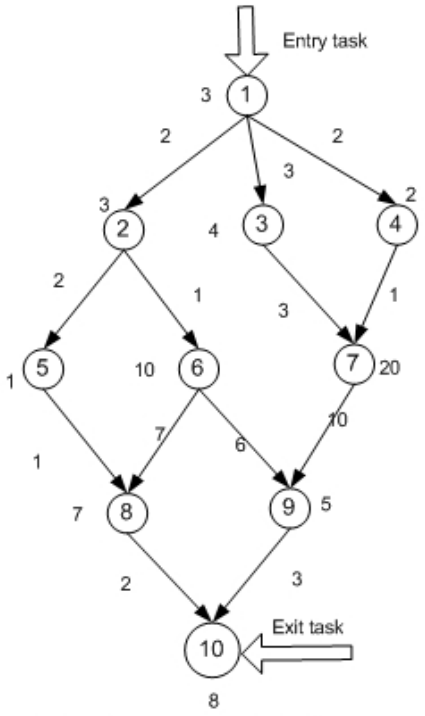
Task Description:

TaskSet {T1, T2, …, T9, T10 }
T1 is the entry task;
T10 is the exit task;
T2, T3 and T4 can not start until T1 finished;
T5 and T6 can not start until T2 finished;
T7 can not start until both T3 and T4 finished;
T8 can not start until both T5 and T6 finished;
T9 can not start until both T6 and T7 finished;
T10 can not start until both T8 and T9 finished;

Estimated execution time for each task:
{T1=3s; T2=3s; T3=4s; T4=2s; T5=1s; T6=10s;
T7=20s; T8=7s; T9=5s; T10=8s;}

Estimated communication cost between tasks:
{T1→T2=2s; T1→T3=3s; T1→T4=2s;
T2→T5=2s; T2→T6=1s; T3→T7=3s;
T4→T7=1s; T5→T8=1s; T6→T8=7s;
T6→T9=6s; T7→T9=10s; T8→T10=2s;
T9→T10=3s;}

(a) A Typical Task Description



(b) DAG Generated by Task Analyzer

Figure 4: Task Analyzer generates DAG

following behind it. The task analyzer will take the user request (usually it contains the necessary task description information) as input and generate DAGs as output. Figure 4 illustrates a typical task description file and the DAG generated by the task analyzer.

5 CONCLUSION

In this paper, we proposed a framework for simulating energy-efficient data grids. First, we proposed a system architecture for data grids. Next, we discussed in detail about the simulation framework and the new features of our framework. The framework is focused on scheduling data-intensive tasks running on a data grid in a way to conserve energy without adversely effecting performance. Specifically, the framework can simulate three scheduling phases, which are geared to make the best tradeoffs between energy savings and performance. The simulation framework can be used to evaluate energy-aware scheduling strategies designed for data grids. As a future research direction, we will make use of the framework to simulate a family of scheduling algorithms.

ACKNOWLEDGMENTS

The work reported in this paper was supported by the US National Science Foundation under Grant No. CCF-0742187, Auburn University under a startup grant, the Intel Corporation under Grant No. 2005-04-070.

REFERENCES

Bryce, R. 2000. Power struggle. Interactive Week, December 2000. <http://www.zdnet.com/intweek/>.

Chase, J. and R. Doyle. 2001. Energy management for server clusters. In *Proceeding of the 8th Workshop Hot Topics in Operating Systems* (HotOS-VIII)165, May 2001.

Cuenca, J., D. Gimenez and J.-P. Martinez, Heuristics for work distribution of a homogeneous parallel dynamic programming scheme on heterogeneous systems. Proc. IEEE Heterogeneous Comp. Workshop, 2006.

W. Dally, P. Carvey, and L. Dennison. 1998. The Avici Terabit Switch/Rounter. In *Proc. Hot Interconnects* 6:41-50.

E.N. M. Elnozahy, M. Kistler, and R. Rajamony. 2000. Energy-efficient server clusters. In *Proc. Int'l Workshop Power-Aware Computer Systems*.

Hagras, T. and J. Janecek. 2006. A high performance, low complexity algorithm for compile-time task scheduling in heterogeneous systems. In *Proc. IEEE Heterogeneous Computing Workshop*.

Kim, S.J. and J.C. Browne. 1998. A General Approach to Mapping of Parallel computations upon Multiprocessor Architectures. In *Proc. Int'l conf. Parallel Processing* 3:1-8, University Park, Penn.

Kishimoto, Y. and S. Ichikawa. 2006. Optimizing the configuration of a heterogeneous cluster with multiprocessing and execution-time estimation. Proc. IEEE Heterogeneous Computing Workshop, 2006.

Lorch J. and A. Smith. 1998. Software Strategies for Portable Computer Energy Management. IEEE Personal Commun. 5:60-73.

Mellanox Technologies Inc. 2004. Mellanox Performance, Price, Power, Volumn Metric (PPPV), <http://www.mellanox.co/products/shared/PPPV.pdf>.

Moore, B. 2002. Taking the data centre power and cooling challenge. *Energy User News*.

Jacob, T. and E.Chen. 1993. Fast algorithms for distributed resource allocation. *IEEE Trans. Parallel and Distributed Sys*. 4(2):188-197.

Pande, S.S., D.P. Agrawal and J. Mauney. 1995. A scalable scheduling method for functional parallelism on distributed memory multiprocessors. *IEEE Trans. Parallel and Distributed Systems* 6(4):388-399.

Qin, X. and H. Jiang. 2005a. Data grids: supporting data-intensive applications. In *Wide Area Networks. High Performance Computing: Paradigm and Infrastructure* 481-494. Edited by L. T. Yang and M. Guo, John Wiley and Sons.

Qin, X. and H. Jiang. 2005b. A dynamic and reliability-driven scheduling algorithm for parallel real-time jobs on heterogeneous clusters. *Journal of Parallel and Distributed Computing* 65(8):885-900.

Qin, X. and H. Jiang, 2006. A novel fault-tolerant scheduling algorithm for precedence constrained tasks in real-time heterogeneous systems. *Parallel Computing* 32(5-6):331-356.

Rabaey, J. and M. Pedram (Editors). 1998. Lower Power Design Methodologies, Kluwer Academic Publisher, Norwell, MA.

Raghunathan, N., K. Jha, and S. Dey. 1998. *High-level power analysis and optimization*. Kluwer Academic Publisher, Norwell, MA.

Ranaweera, S. and D.P. Agrawal. 2000. A task duplication based scheduling algorithm for heterogeneous systems. In *Proc. Parallel and Distributed Processing Symp.* 445-450.

Siegel, H.J. et al. 2006. Mapping subtasks with multiple versions on an ad-hoc grid. In *Proc. IEEE Heterogeneous Computing Workshop*.

Sih, G.C. and E.A. Lee. 1993a. Declustering: a new multiprocessor scheduling technique. *IEEE Trans. Parallel and Distr. Sys.* 4(6):625-637.

Sih, G.C. and E.A. Lee. 1993b. A compile-time scheduling heuristic for interconnection-constrained heterogeneous processor architectures. *IEEE Trans. Parallel and Distr. Sys.* 4(2):175-186.

Shin, Y. and K. Choi. 1999. Power conscious fixed priority scheduling for hard real-time systems. In *Proc. Design Automation Conf.*.

Srinivasan, S. and N.K. Jha. 1999. Safety and reliability driven task allocation in distributed systems. *IEEE Trans. Parallel and Distributed Systems* 10(3):238-251.

Srivastava, M., A. Chandrakasan And R. Brodersen. 1996. Predictive system shutdown and other architectural techniques for energy efficient programmable computation. *IEEE Trans. VLSI Systems*. 4(1):42-55.

Woodside, C.M. and G.G. Monforton. 1993. Fast allocation of processes in distributed and parallel systems. *IEEE Trans. Parallel and Distributed Sys.* 4(2):164-174.

M.Y. Wu and D.D. Gajski, "Hypertool: A Performance Aid for Message-Passing Systems," IEEE Trans. Parallel and Distr. Sys., vol. 1, no. 3, pp. 330-343, July 1990.

Yajnik, S., S. Srinivasan, and N.K. Jha. 1994. TBFT: a task-based fault tolerance scheme for distributed systems. In *Proc. Int'l Conf. Parallel and Distributed Computer Sys*.

Yao, F., A. Demers, and S. Shenker. 1995. A scheduling model for reduced CPU energy. In *Proc. IEEE Annual Foundations of Computer Sci*.374-382.

Zong, Z.-L., A. Manzanares, B. Stinar, and X. Qin. 2006. Energy-efficient duplication strategies for scheduling precedence constrained parallel tasks on clusters. In *Proc. Int'l Conf. Cluster Computing*.

AUTHOR BIOGRAPHIES

ZILIANG ZONG received his B.S. and M.S. degrees in Computer Science from Shandong University, China, in 2002 and 2005 respectively. He is currently a Ph.D. candidate in the Department of Computer Science and Software Engineering at Auburn University. During Oct. 2003 – Oct. 2004, he studied as research assistant student in the Artificial Intelligence Lab of Toyama University, Japan. He is an IEEE student member and a recipient of 2006 IEEE Technical Committee on Scalable Computing (TCSC) Student Travel Award. His research interests include energy-efficient scheduling, high performance computing, distributed storage systems and real-time communication and embedded systems. His web page can be found via <www.auburn.edu/~zzz0003>

KIRANMAI BELLAM received her BS in Computer Science from the University of Madras, Chennai, India, in 2004. She received the MS degree in Computer Science from New Mexico Institute of Mining and Technology in 2006. She is currently a Ph.D. student in the Department of Computer Science and Software Engineering at Auburn University. Her research interests include real-time systems, fault tolerance, storage system, energy-efficient computing, and real-time systems.

XIAO QIN received the BS and MS degrees in Computer Science from Huazhong University of Science and Technology, China, in 1996 and 1999, respectively. He received the PhD in Computer Science from the University of Nebraska-Lincoln in 2004. Currently, he is an Assistant Professor of Computer Science at Auburn University. Prior to joining Auburn University in 2007, he had been with New Mexico Institute of Mining and Technology for three years. In 2007, he received an NSF Computing Processes & Artifacts (CPA) Award. His research interests include parallel and distributed systems, real-time computing, storage systems, fault tolerance, and performance evaluation. He had served as a subject area editor of *IEEE Distributed System Online* (2000-2001). He has been on the program committees of various international conferences, including IEEE Cluster, IEEE IPCCC, and ICPP. His web page can be found via <www.eng.auburn.edu/~xqin>

YIMING YANG received the BS and MS degrees from Tsinghua University, China, in 1984 and 1987, respectively. He received the PhD in Mechanical Engineering from Washington State University in 1993. Currently, he is a senior software engineer at the Intel Corporation. Prior to joining Intel, he was with Los Alamos National Laboratory. His research interests are software engineering, embedded systems, and high-performance computing.

XIAOJUN RUAN received his B.S. degree in Computer Science from Shandong University, China, in 2005. Currently, he is a Ph.D. student in the Department of Computer Science and Software Engineering at Auburn University. Before joining Auburn University, he was a MS student at New Mexico Institute of Mining and Technology. His research interests are storage systems, cluster computing, and energy-efficient computing.

ADAM MANZANARES received his B.S. degree in Computer Science from New Mexico Institute of Mining and Technology in 2006. Currently, he is a Ph.D. student in the Department of Computer Science and Software Engineering at Auburn University. Prior to August 2007, he was a MS student at the University of Colorado at Boulder. His research interests focus on storage systems, cluster computing, distributed systems, and wireless networks.